APRENDIZADO DE MÁQUINA

REGRESSÃO LOGÍSTICA (IMPLEMENTAÇÃO SIMPLES)

PROF. RONALDO CRISTIANO PRATI
RONALDO.PRATI@UFABC.EDU.BR

BLOCO A, SALA 513-2

SIGMOIDE

```
def sigmoid(z):
    return(1 / (1 + np.exp(-z)))

def predict(theta, X, threshold=0.5):
    p = sigmoid(X.dot(theta.T)) >= threshold
    return(p.astype('int'))
```

FUNÇÃO DE CUSTO

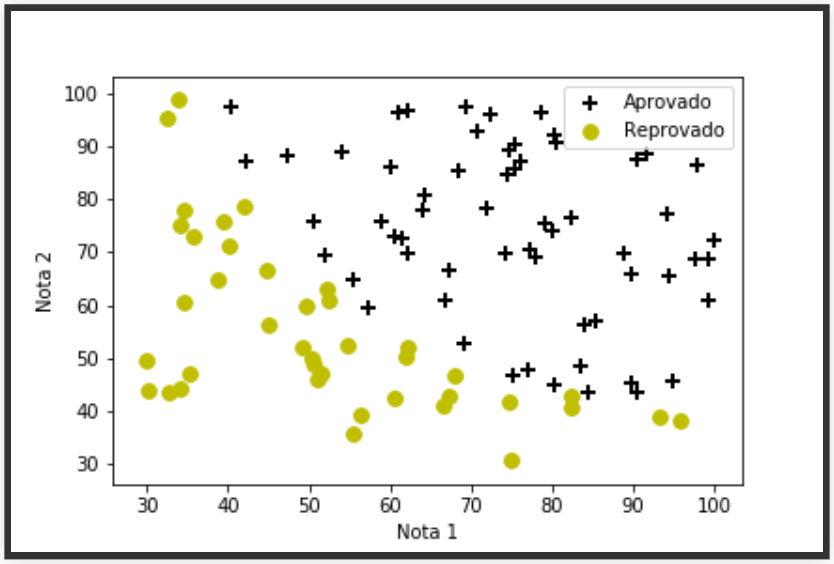
```
def costFunction(theta, X, y):
    m = y.size
    h = sigmoid(X.dot(theta))

    J = -1*(1/m)*(np.log(h).T.dot(y)+np.log(1-h).T.dot(1-y))

    if np.isnan(J[0]):
        return(np.inf)
    return(J[0])
```

DADOS

- Notas de duas avaliações
- Aprovação segue regra não linear



EXECUÇÃO

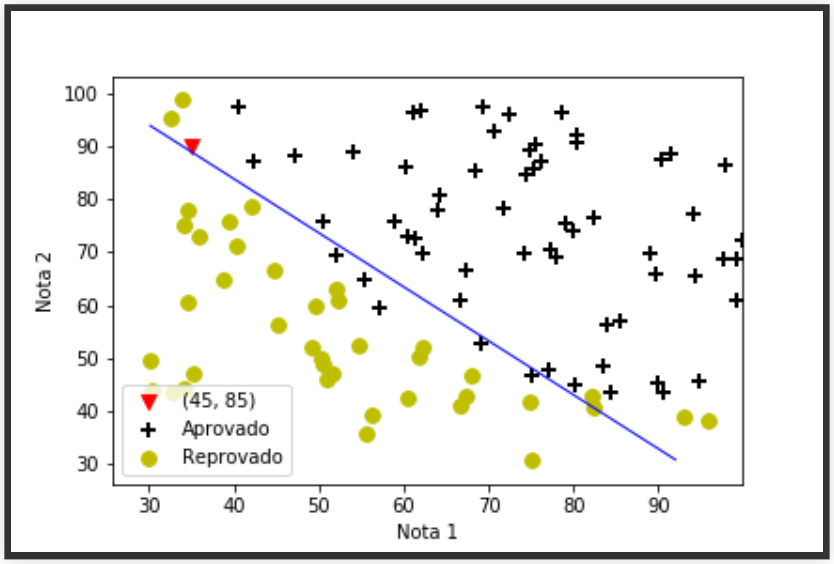
```
import pandas as pd
import numpy as np
from scipy.optimize import minimize

data = np.array(pd.read_csv("ex2data1.txt",sep=",",header=None))

X = np.c_[np.ones((data.shape[0],1)), data[:,0:2]]
y = np.c_[data[:,2]]
initial_theta = np.zeros(X.shape[1])
cost = costFunction(initial_theta, X, y)
grad = gradient(initial_theta, X, y)
print('Cost: \n', cost)
print('Grad: \n', grad)
res = minimize(costFunction, initial_theta,
    args=(X,y), method=None, jac=gradient, options={'maxiter':400})
```

FRONTEIRA DE DECISÃO LINAER

- Notas de duas avaliações
- Aprovação segue regra não linear



TERMOS QUADRATICOS

- Incorporando termos quadráticos aos dados

```
X = np.c_[np.ones((data.shape[0],1)), data[:,0:2], data[:,0:2]**2 ]
y = np.c_[data[:,2]]
initial_theta = np.zeros(X.shape[1])
cost = costFunction(initial_theta, X, y)
grad = gradient(initial_theta, X, y)
print('Cost: \n', cost)
print('Grad: \n', grad)
res = minimize(costFunction, initial_theta,
    args=(X,y), method=None, jac=gradient, options={'maxiter':400})
```

FRONTEIRA DE DECISÃO QUADRÁTICAS

- Notas de duas avaliações
- Aprovação segue regra não linear